Safety evaluation and management of complex systems: A system engineering approach

Romaric Guillerm[1, 2], Hamid Demmou[1, 2], Nabil Sadou[3]

[1] LAAS-CNRS ,
7 avenue du Colonel Roche, F-31077 Toulouse, France
[2] Université de Toulouse; UPS,
F-31062 Toulouse, France
{guillerm, demmou}@laas.fr,

[3] SUPELEC / IETR
Avenue de la Boulais, F-35511 Cesson-Sevigne
nabil.sadou@supelec.fr

Abstract. This paper addresses the problem of safety evaluation of complex systems. It proposes an original and rigorous approach that integrates safety analysis in system engineering processes. The approach is based on system engineering (SE) principles and uses the famous industrial SE standard ANSI/EIA-632. The objective is to help designers and safety engineers in safety management of complex systems. For an efficient design, the model driven design is adopted through the definition of an information model. The system language *SysML* (System Modeling Language) is used to address requirements definition and their traceability towards the solution and the Verification and Validation (V&V) elements. This common language allows sharing information between the different persons involved in the design project like system engineer and safety engineer.

Keywords: system engineering, safety, requirements, EIA-632, SysML.

1 Introduction

Complex systems are systems with numerous components, interconnections, interactions or interdependencies that are difficult to describe, understand, predict, manage, design, and/or change (Magee and de Weck, 2004). In these systems, a change in one part may have effects on other parts of the system.

Most of modern systems are inherently complex (Bar-yam, 2005). This complexity makes the systems engineering processes (including acquisition and supply, technical management, system design and technical evaluation processes) more critical and difficult. So, performing of important system level proprieties such as reliability, safety and security (Avizienis and al, 2004) become difficult. Indeed, safety of complex systems relies heavily on the emergent properties (Levenson, 2004), (Black and Koopman, 2009) that result from the complex interdependencies that exist among the involved systems and their environments. It is necessary to address safety properties in an overall study, early in the design phase.

For an effective analysis, safety processes have to consider some constraints. The most important ones, listed in (Guillerm and al, 2010), are:

1. The emergent aspect of safety properties imposes to consider safety not only in the small but also in the large (at system level).
2. Different persons involved in the project need to work with different views of the system (e.g. systems engineer's view, safety engineer's view). It is necessary to guaranty the consistence of these different views.
3. Requirement engineering is a critical process in system engineering. So, safety requirements and their traceability need a particular attention.

Taking into account these constraints allows to improve complex system development and to overcome the limitation and weakness (Rasmussen, 1997) of the actual safety evaluation processes.

Requirements engineering (RE) (Sommerville, 2006) is a critical system engineering process (Juristo et al, 2002). It is a key problem area in the development of complex systems (Brooks, 1987). A common classification defines two types of requirements; functional and non-functional (Robertson and Robertson, 2006). Functional requirements describe the services that the system should provide. Non-functional requirements are related to emergent system properties (such as safety) and cannot be attributed to a single system component. This shows the necessity of a global approach for safety management, from the requirements definition to the system verification and validation, which is fundamental for the success of the system design project.

To deal with these different aspects, we propose a global approach for safety analysis. Indeed, safety must be addressed as a global property and safety requirements (Gotel and Finkelstein, 1994) must be formulated not only in the small (sub-system level) but in the large (system level). Two aspects of requirements engineering are considered: the requirements development (including elicitation (Goguen and Linde, 1993), documentation, analysis and validation processes), and the requirements management (including maintainability management, changes management and requirements traceability processes (Parviainen and al, 2004)).

A literature review shows some works that attempt to address the safety evaluation of complex system. For example, safety standards (as the ARP-4754) are useful to understand activities related to safety evaluation, but they don't define a unified approach with the nominal conception activities. This means that safety activities are not integrated in the system engineering processes. Recent projects like ESACS (Bozzano & al., 2003), ISAAC (Akerlund, 2006), ASSERT (Conquet, 2008) which deal with safety of complex systems don't provide a real global approach. They are focused on methods and tools that can be used in safety evaluation. In other works like those presented in (Leveson & al., 2007), in addition to the fact that safety aspects are separated from the design activities, the proposed methodology is clearly oriented to software domain.

This paper presents two aspects of our work on safety evaluation of complex system. The first part concerns the integration of safety management in system engineering process. The objective is to help engineers by proposing a new approach based on system engineering best practices. It uses the recommendation of the system engineering standard EIA-632 (Guillerm and al, 2010).

The second part presents an information model based on *SysML* language (Sanford and al, 2009) to address requirements definition and their traceability (Gotel and Finkelstein, 1994), (Sahraoui, 2005) towards the solution elements and the V&V (Verification and Validation) elements. Safety requirements are integrated on RE activities, including management activities related to maintenance, and traceability.

The paper is structured into five sections. The second section introduces the design framework. The integration approach is then presented in the third section. In the fourth section, the information model is proposed for efficient management of safety requirements. The last section gives some conclusions and future works.

2 The systems engineering framework for complex system development

System engineering is a methodical and disciplined approach for the design, realization, technical management, operations, and retirement of a system. It is a collaborative and interdisciplinary process of resolution of problems, supporting knowledge, methods and techniques resulting from the sciences and experiment for defining a system. System engineering concepts are adequate specifically for complex problems; research issues undergone can bring a solution (Sahraoui and al, 2004).

In this part, we introduce some concepts of system engineering and the standard EIA-632 which are the basis of our safety global approach.

2.1 System engineering concepts

System engineering (SE) is the application of scientific and engineering efforts to transform an operational need into a design solution, with a description of system performance parameters and system configurations, through an iterative process of definition, synthesis, analysis, design, test and evaluation. SE is an interdisciplinary approaches that:

1. Encompasses the scientific and engineering efforts related to the development, manufacturing, verification, deployment, operations, support and disposal of systems products and processes.
2. Develops needed user training, equipment, procedures and data.
3. Establishes and maintains configuration management of the system.
4. Develops work breakdown structures and statements of work, and provides information for management decision-making.

SE is an appropriate combination of methods and tools for suitable methodological process and systems management procedures.

We distinguish three levels in SE, as illustrated in Figure 1.

The first level, “SE processes”, focus on high-level issues, high-level requirements such as business needs and strategic needs and methods.

The second level, “SE methodologies and methods”, deals with all technical issues such as systems requirements and design methodologies.

The third level, “SE tools or technologies”, covers the implementation issues concerning the tools to be used, the required technologies to respond to the various assets of requirements such as reliability costs, maintainability and enabling technologies.

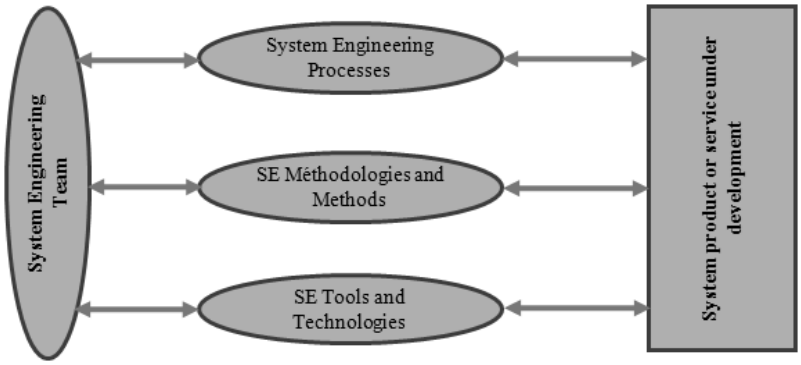

Fig.1. Three levels of system engineering.

These entities, such as processes, methods and tools, are the conceptual basis of our approach taken from SE best practice. In the first step, the processes can be identified with respect to the accumulated know-how. The second step concerns the methods to be used. The methods can be either developed or may be existing methods. Implementing the process, one method cannot be chosen for its flexibility or popularity, but only if it reflects the semantics of the process. No taxonomy has yet been developed for corresponding processes and methods. The third step concerns the tools that do not correspond to the processes but to the methods; hence in this approach we cannot use a tool to implement a process without first identifying the associated methods.

2.2 EIA-632 standard

The beginnings of systems engineering can be traced back to the Bell Telephone Laboratories in the 1940s (Auyang, 2004). Thirty years later, the first U.S. military standard was published (MIL-STD-499A, 1969.). It is focused on systems engineering, providing the first definition of the scope of engineering management. Nowadays, the standard ANSI/EIA-632 “Processes for Engineering a System”, which provides a typical systems engineering Work Breakdown Structure (Valerdi and Wheaton, 2005), is one famous standard, currently used in the industrial and military fields. This standard covers the product life-cycle from the needs capture to the transfer to the user. The processes are well described by the following Electronic Industries Alliance (EIA) standard (Figure 2). It is constituted by 13 processes (EIA-632, 1999) covering the management issues, the supply/acquisition, design and requirement, verification and validation processes:

1. Technical management processes (three processes): these processes monitor the whole process ranging from the initial idea to building a system until the delivery of the system.
2. Acquisition and supply processes (two processes): these processes ensure the supply and acquisition (and are very close to logistics).
3. System design processes (two processes): these processes deal with the elicitation and the acquisition of requirements and their modelling, the definition of the logical design and its physical solution.
4. Product realisation processes (two processes): these processes deal with the implementation issues of the system design and its use.
5. Technical evaluation processes (four processes): these processes deal with verification, validation and testing issues.

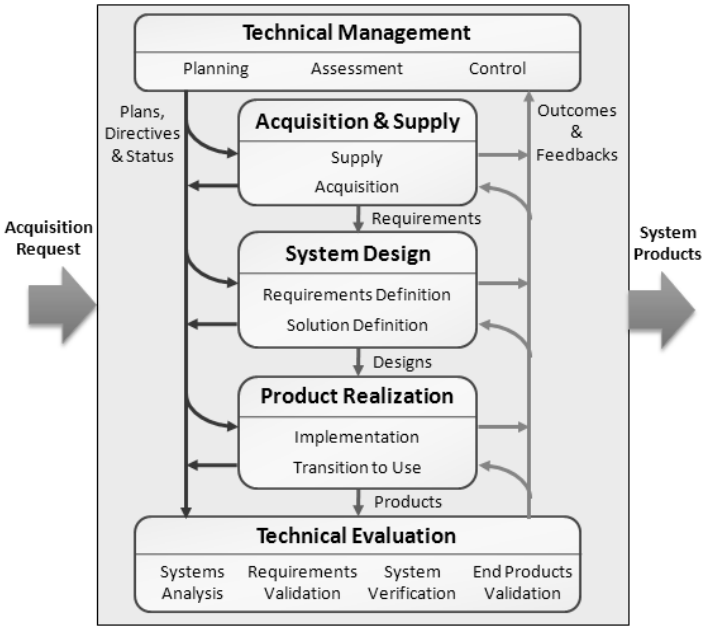

Fig.2. System engineering processes (source EIA-632)

In the work presented in this paper, we focus on the system design process and the technical evaluation process. Note that other processes will be considered in future works.

3 Integration approach

As said in the introduction, safety of complex systems relies heavily on the emergent properties. System Engineering is an ideal framework for the design of complex systems and the need for systems engineering arose with the increasing complexity of systems and projects.

This work for safety integration in SE processes (system engineering approach) is based on the assumption that the safety property can only be treated adequately in their entirety, taking into account all variables, social and technical aspects (Kotovsky and al, 1985). This basis for system engineering has been stated in the principle that a system is more than the sum of its parts. The starting point of the work presented in this paper is the following note provided in EIA-632 standard:

Note: *Standard does not purport to address all safety problems associated with its use or all applicable regulatory requirements. It is the responsibility of the user of this Standard to establish appropriate safety and health practices and to determine the applicability of regulatory limitations before its use.* (EIA-632, 1999).

So, following this note, this section provides the approach that aims to guide designers in addressing safety problems. It describes for each process, how the safety must be considered.

Note that, in reference to SE chain, the proposed approach is illustrated in term of process, which must be defined independently to methods and/or tools. Other projects are focused on methods and tools (see (Akerlund, 2006) and (Bozzano and al, 2003) for example).

The Safety management must follow all steps (processes) of SE from the requirements definition to the verification and the validation of the system. This paper is focused only on:

- *System Design processes* in which we address requirements definition processes and the solution definition processes.
- *Technical Evaluation processes* with the system analysis process, the requirements validation process and the system verification process.

The implementation of the approach consists in identifying and indicating how safety must be considered for each sub-processes of EIA-632.

3.1 System design processes

The *System Design Processes* are used to convert agreed-upon requirements of the acquirer into a set of realizable products that satisfy acquirer and other stakeholder requirements (EIA-632, 1999). Two processes are involved: the *Requirements Definition process* and the *Solution Definition process*. The interaction between these processes is shown in Figure 3.

3.1.1 Requirement definition process

The objective of the *Requirements Definition Process* is to transform the stakeholder requirements into a set of technical requirements. For functional and non-functional requirements, if this distinction is not possible at the requirement elicitation process level, an analysis may be done in order to categorize requirements. Two types of requirements are defined: the *Stakeholder Requirements* and the *System Technical Requirements*.

Concerning *Stakeholder Requirements*, the developer shall define a validated set of acquirer requirements for the system, or portion thereof.

Generally, safety requirements correspond to constraints in the system. It is necessary to identify and collect all constraints imposed by acquirer to obtain a safe system. A hierarchical organization associates weight to safety requirements, following their criticality.

For *Technical Requirements*, the developer shall define a validated set of system requirements from the validated sets of *stakeholder requirements*. For safety requirements, the system technical requirements traduce system performances. It consists on defining safety attributes (Determine risk tolerability, SIL level, MTBF, MTTR for example).

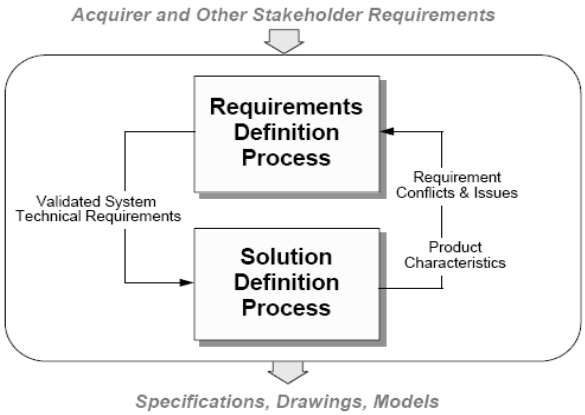

Fig.3. EIA-632 System design process.

Safety requirements can be derived from different sources. The first source is the *stakeholders.* In this case classical requirements elicitation methods can be used (Coulin and al, 2005). The second source is constituted by standards which guide designer to define safety requirements. For example, safety critical systems within the civil aerospace sector are developed subject to the recommendations outlined in (ARP-4754, 1996) and (ARP-4761, 1996). These standards give guidance on the 'determination' of requirements, including requirements capture, requirements types and derived requirements. The third source is outputs of risk analysis. The identified hazards are evaluated in terms of likelihood and severity by means of risk assessment. Architectural design decisions are then made upon choosing appropriate risk mitigation strategies or actions, and safety requirements are defined in response to the chosen mitigation mechanisms (Wu and Kelly, 2006).

When requirements are defined, it is possible to define some attributes to facilitate their management, for example with an expression of requirements using *SysML* (SysML, 2006), (Bock, 2006) or *UML 2* (Friedenthal and Kobryn, 2004). It is possible to link requirements to the design solution. This point will be presented in the 4th section.

3.1.2 Solution Definition Process

The *Solution Definition Process* is used to generate an acceptable design solution.

For *Logical Solution Representations*, the developer shall define one or more validated sets of logical solution representations that conform with the technical requirements of the system. Formal or semi-formal models (*UML*, *SysML*, Petri net, finite-state machine…) are recommended in this process for the solution modeling. The use of formal methods allows the automation of verification and analysis. In this processes, safety analysis techniques will be used to determine the best logical solution.

The *Physical Solution Representations* are derived from logical solution representation and must respect all requirements, particularly safety requirements. The same safety analysis may be done when the physical solution representation is available. The same recommendations than for logical solution remain true. However, when approaching the physical solution, it appears that the semantics of *SysML* does not seem quite complete. So it is possible to use architecture languages such as *AADL* (Aer, 2004), *VHDL-AMS* (Verries and al, 2008) (Vhd, 1999), *SystemC-AMS*…

As this design process progresses, details of the system are obtained. So, the hazards analysis will be refined and new hazards may emerge. The Chosen mitigations may themselves bring new safety problems. Therefore, new risk mitigation actions may need to be identified and safety requirements refined. The process evolves (see System design process) until all identified hazards have been mitigated sufficiently in an acceptable manner.

3.2 Technical Evaluation Processes

The *Technical Evaluation Processes* are intended to be invoked by one of the other processes for engineering a system. Four processes are involved: *Systems Analysis*, *Requirements Validation*, *System Verification*, and *End Products Validation*. The relationship between these processes is shown in Figure 4.

3.2.1 System Analysis Process

The *Systems Analysis Process* is used to: (1) provide a rigorous basis for technical decision making, resolution of requirement conflicts, and assessment of alternative physical solutions; (2) determine progress in satisfying system technical and derived technical requirements; (3) support risk management; and (4) ensure that decisions are made only after evaluating the cost, schedule, performance, and risk effects on the engineering or reengineering of the system (EIA-632, 1999).

In this process safety is involved at each item. Indeed, (1) in the case of requirement conflict, higher priority requirements must be safety ones. These requirements are used in the assessment of alternative physical solutions. (2) The designer has to determine the satisfaction of system technical requirements and derived technical safety requirements. The interest of using semi formal model for requirement traceability analysis becomes obvious. (3) The risk management is used to develop risk management strategies of the design project. (4) Safety aspect can generate additional cost. The designer has to evaluate this cost.

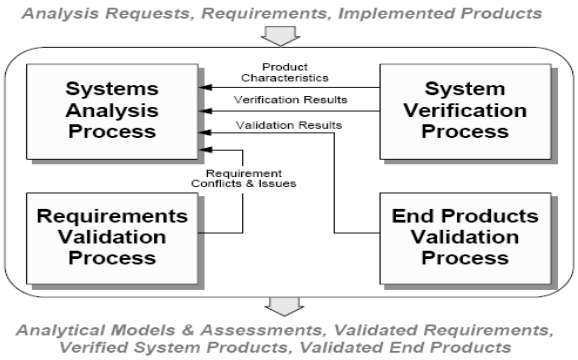

Fig.4. Technical evaluation processes

3.2.2 Requirements Validation Process

Requirements Validation is critical to successful system product development and implementation. Requirements are validated when it is certain that they describe the input requirements and objectives such that the resulting system products can satisfy them.

In this process, a great attention is given to traceability analysis, which allows verifying all the links among *Stakeholder Requirements*, *Technical and Derived Technical Requirements*, and *Logical Solution Representations*. Like other requirements, safety requirements must be validated.

Frequently, specifications are written in natural language and are the base of the design, test and validation phases. The natural language can be ambiguous. The consequence is that the requirements can be differently interpreted. It is thus necessary to propose a method of progressive refinement of the requirements towards models with known semantics allowing their formalization.

To facilitate the step of requirements validation, semi-formal solutions, like *UML* (Booch and al, 1998) or *SysML* (Bock, 2006) can be used for good formulation of requirements. Indeed, the diversity of people concerned by design project can have limited knowledge concerning the structure of the future system, that's why industry-scale requirement engineering projects are so hard. Thus, the *UML* or *SysML* languages, with their different diagrams, can be helpful.

3.2.3 System Verification Process

The *System Verification Process* is used to ascertain that the generated system design solution is consistent with its source requirements, in particular, safety requirements.

In case of safety verification, when the agreement between designers and safety teams is obtained, a last safety analysis called System Safety Assessment (SSA) is done to consolidate the product from the safety point of view. Some traceability models allow defining the procedure of verifying safety requirement. These procedures are planned at the definition of safety requirement. Simulation (Zeigler and al, 2000) is an appropriate method that can be used to achieve system verification. Other methods like test, virtual prototyping or model checking are also used.

4 Information model

Requirements engineering is an important phase in a system design. It is important to perform it correctly for project success (Juristo et al, 2002). This activity is difficult in case of complex systems. Requirements formalization can help the designers in their requirements engineering activities. The introduction of models can be useful to achieve the formalization task. Model-driven engineering, in which models are the main artifact during system development, is an emergent approach that tries to address system complexity by the intensive use of models. In this part we present an information model based on *SysML* to properly define and manage requirements and particularly safety requirements.

4.1 Requirements management

In complex systems engineering, an important number of documents is produced specially during the system definition phase. Statistical studies show that the success of a project depends strongly on the definition and the management of requirements. Requirements management allows to collect requirements, to facilitate their expression, and to validate them. It must also ensure that each requirement is properly declined, allocated, monitored, satisfied, made verifiable, verified and justified.

Figure 5 presents an overview of the requirements relationships as defined in the EIA-632 standard. The proposed information model follows this pattern. We see that *other stakeholder requirements*, when added to the *acquirer requirements*, make up a set of stakeholder requirements that are transformed into *system technical requirements*. The *logical* and *physical solution representations* are derived from *technical requirements*. *Design solution* and *specified requirements* are defined by completing the *solution definition* Process.

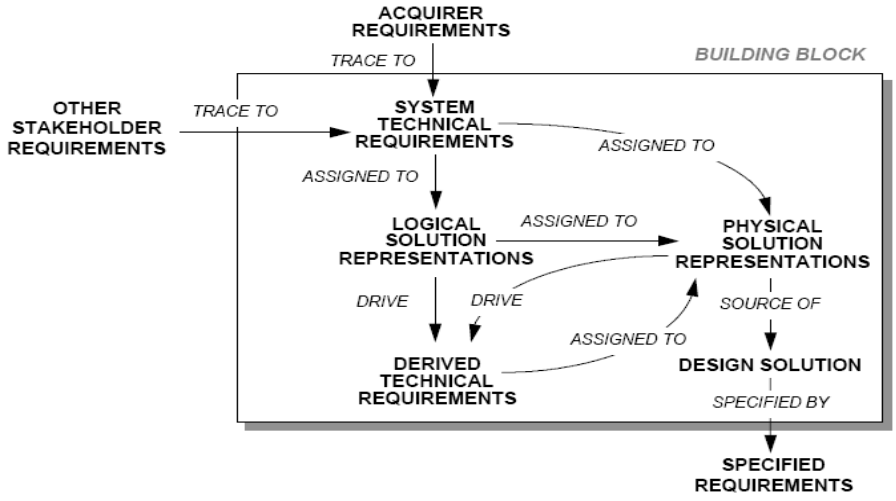

Fig.5. Requirements relationships: EIA-632 model.

4.2 Role of the information model

The information model can be considered as a database used to share and capitalize knowledge. It is accessible for the persons involved in the design project and aims to guide the design process. It is also used to manage requirements changes, and then evaluating the project progress.

When different views exist in the design team as it is often the case (design engineer's view, safety engineer's view for example) the information model makes the system more understandable as it is based on a common language.

When transforming needs into system definition, modeling can help to properly achieve this task. Indeed, during this transformation, we will gradually go from abstract concepts to a rigorous definition of the system.

In modeling, there are 2 separate areas: the problem area and the possible solutions area. At the beginning of the project, the representation of the problem area is more important than the representation of the possible solutions area. During the progress of the design, representation of possible solutions area will be enriched to achieve the strict definition of the system. In parallel the overall representation of the problem area will be enriched to better define the expectations of the system (needs/requirements) and will stabilize itself. The transition between the problem domain and the solution domain is a very delicate point of system engineering. It must be expressed by allocating requirements/properties/constraints on possible solutions. These allocations will generate traceability links, which are crucial for the system verification and validation activities. We propose an information model, based on *SysML* language that will be compatible with the requirements management of the standard EIA-632, to takes into account safety and risk management aspects.

4.3 Systems Modeling Language (SysML)

SysML is a systems modeling language that supports specification, analysis, design, verification and validation of a broad range of complex systems. The language is an evolution of *UML* 2.0 and is defined for systems that may include hardware, software, information, processes and personnel. It aims to facilitate the communication between heterogeneous teams (mechanical, electrical and software engineers for instance). The language is effective in specifying requirements, architecture, and behavior. It allows the allocation of elements to models and the definition of constraints on system properties to support engineering analysis. *SysML* allows the modeling and supports different views:

- The requirements view: using requirements diagram and use case diagram,
- The structure view: based on block diagram (internal/external),
- The behaviour view: using *statechart* diagram, activity diagram, and sequence diagram,
- The constraints view: with parametric diagram.

SysML seems to be an excellent candidate for a common language. It allows sharing specifications of a complex system between different trades, between design engineers and safety engineers in our case. *SysML* allows expressing the requirements using the requirements diagram. It also defines some relationships that link a given

requirement to other requirements or elements of the model. With SysML, it is possible to have:

- A definition of hierarchy between requirements,
- requirement derivation,
- requirement satisfaction by a model element,
- requirement verification by a test case (*TestCase*) or
- requirement refinement.

With *SysML* we can create traceability links between requirements and system components. These links allow performing impact analysis of requirements change or modification. Thus, it is possible to assess the consequences of a requirement change on the system safety using the links defined between requirements, functions and components.

4.4 *SysML* Extension

The *SysML* language proposes basic concepts (like requirements or blocks) for which some attributes are associated. The Semantic of *SysML* is enriched through the extension mechanism, in order to adapt it to our approach. We define more detailed requirement element, several types of requirements, and also additional traceability links.

4.4.1 Enriched requirement

The concept of requirement is fundamental in system design. In *SysML* a specific diagram is dedicated to requirements with a possibility to define traceability links for requirements. The objective is to guide the development (by linking needs to the design solution) and to facilitate the verification and validation phase or the management of requirements changes. Figure 6 shows *SysML* stereotype requirement definition. Requirement in *SysML* is composed by two attributes: an identifier (Id) and the description of the requirement (Text).

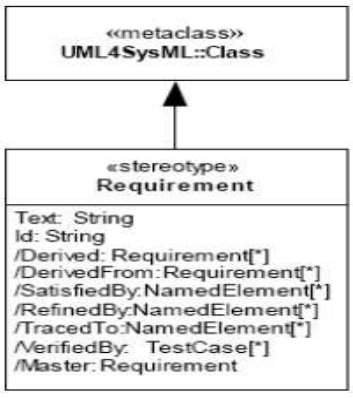

Fig.6. The requirement stereotype in *SysML*.

The definition of stereotype requirement is enriched in order to adapt it for our need. This is done by adding some new attributes that are inspired form the RPM profile (Requirement Profile for MeMVaTEx) (Albinet and al, 2008), the recommendation of AFIS (the French association of system engineering) (AFIS, 2005) and the Snow Card Volere (Roberston, 2010). As shown in the Figure 7, these attributes are:

- Category: shows if the requirement is functional or non-functional (reliability, availability, safety, performance, cost or maintainability…),
- Justification: description of the justification of the requirement (why this requirement?),
- Source: acquirer, other stakeholder, standard, regulatory authorities, risk analysis,
- Target: end product or enabling products (development, production, test, deployment, training, support or disposal product…),
- Maturity: to inform about the life phase (or status) of the requirement: original, defined, validated or verified,
- Criticality: low, medium or high, used to characterize the importance of the requirement in terms of risk,
- Flexibility: low, medium or high,
- Priority: low, medium or high, used to differentiate a mandatory requirement from an optional one.

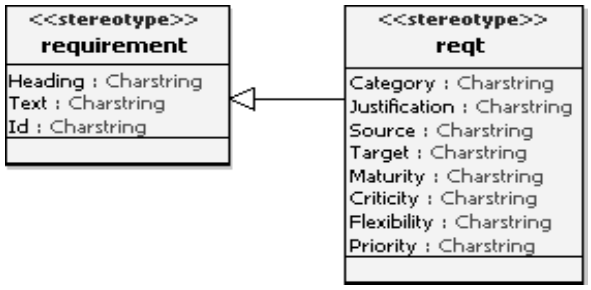

Fig.7. Proposed requirement model

4.4.2 New requirement Stereotypes

Stereotypes allow extending the semantic of SysML. They allow defining new types of blocks.

We define new stereotypes for requirements (see Figure 8). These stereotypes are based on the classification of requirements in system engineering (particularly in the standard EIA-632, see figure 5). So, it is possible to create and define requirements of "*acquirerReqt*", "*otherStakeholderReqt*", "*systemTechnicalReqt*" or "*specifiedReqt*" stereotype. These new requirements stereotypes facilitate the distinction between the different types of requirements.

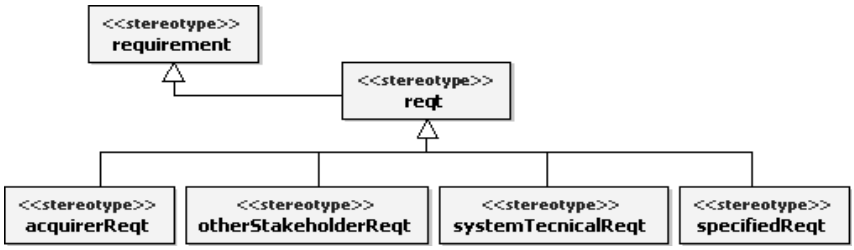

Fig.8. Requirements stereotypes

4.4.3 Risk stereotype

Concerning safety requirements, which can be derived from risk analysis, a block risk is defined and is linked to safety requirements (see figure 9). In fact, identification of risks is the starting point for many studies about safety/reliability. Thus, defining a block "*risk*" in the information model and linking it to the safety requirements, allows to improve the comprehension of the presence of some safety requirements and to justify them. The ultimate objective is to improve the safety analysis of the system.

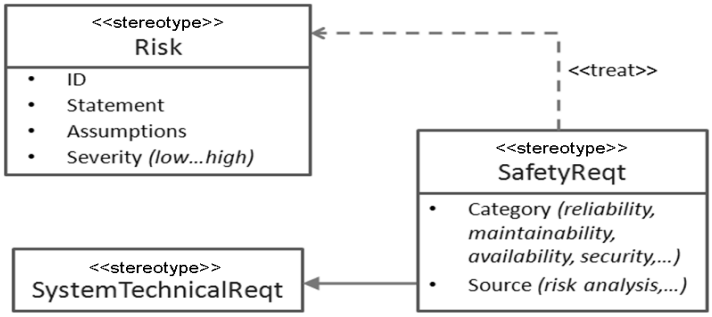

Fig.9. From risk to safety requirements

4.5 Presentation of the information model

As said previously, the information model is based on *SysML* and follows the SE process (EIA-632 standard). It is considered as the "system" knowledge basis of the design project, allowing data sharing between all different trades (mechanical, hydraulic, thermal, electrical...). Therefore, it is intended to model the "system" level, showing the interactions between the system and its environment and also the connections between subsystems. The 3 basic concepts of system design are: requirements, design solution and V&V (Validation and Verification). All these concepts are included and represented in the information model.

Safety authorities impose a separation of system design concepts (requirements, design solution and V&V). They must be developed independently. From this

observation, the proposed approach allows the expression of these concepts, with a clear separation between them. However Traceability mechanisms are provided to link these concepts in order to facilitate impacts analysis. We propose a way to use *SysML* that allows structuring the elements of the design project with respect to the concepts separation (see Figure 10). In other words, our approach allows a rigorous organization of the project design. Indeed, different diagrams manage different concepts.

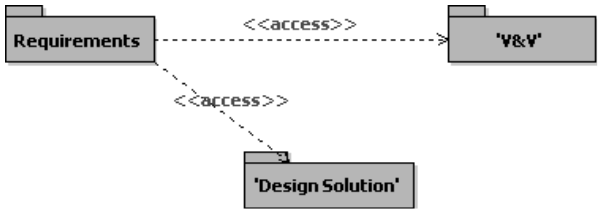

Fig.10. Package diagram of concepts separation

In the information model (Figure 11), we can see that all types of requirements are represented. We see that they follow the EIA-632 standard recommendations. All traceability links required by the EIA-632 are present in this model, and the distinction between logical solution (functional part) and physical solution (component part) appears.

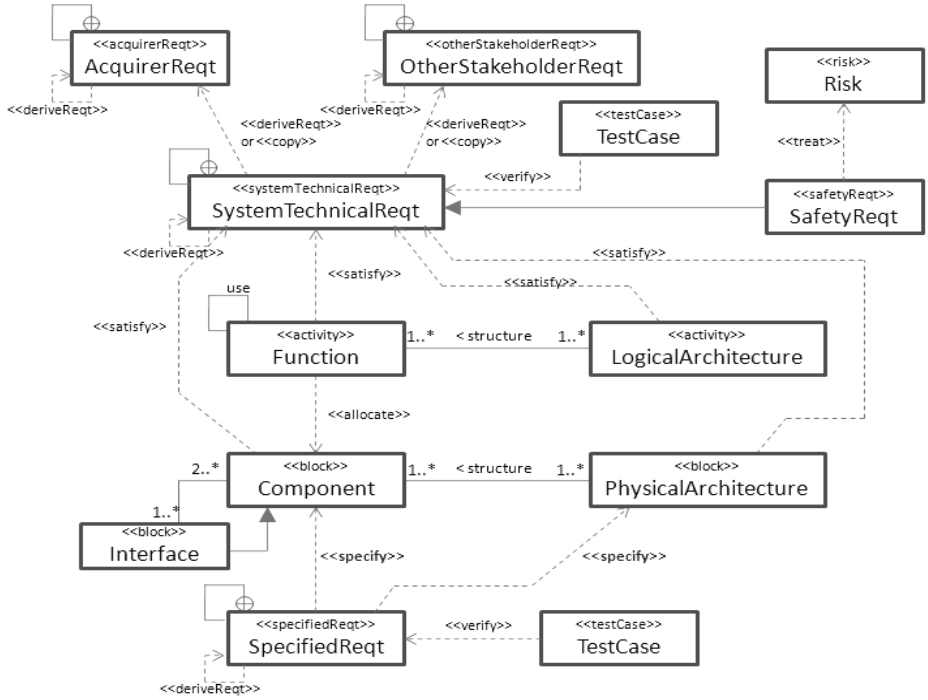

Fig.11. Information model

In this model, we highlight the definition of interfaces, which are components themselves and which links several components together. The concept of interface is essential for a proper system design. Indeed, some problems encountered during development can result from a bad definition of interfaces.

The last important element included in this model, other than requirement and solution element, is the "*TestCase*". These elements of V&V are included in the model to be directly connected to the requirements that must be satisfied by the *TestCase*.

5. Conclusion

In this paper we presented a new approach for safety evaluation of complex systems. The contribution consists of two main points. The first one concerns the integration of safety in system engineering processes. The proposed methodology is based on EIA-632 system engineering standard and aims to help engineers involved in a design project to manage the safety aspect of the system in parallel to other design activities. It allows an explicit consideration of safety in systems engineering process by defining the specific activities related to safety. All system engineering processes, from requirements definition process to verification and validation process, are concerned by the integration approach. As the approach is based on the EIA-632 standard and follows the different processes and sub-processes of this one, it can be incorporated easily into systems engineering concepts. This gives a framework for safety management showing that the SE concepts are adequate, specifically for complex problems.

Requirements engineering is a crucial activity in the design of complex system. The second contribution of the paper is the proposition and the definition of an information model based on the *SysML* language. It is done by an extension of this language by adding new stereotypes and new attributes to requirements. We also defined new links (*specify* and *treat*) between specified requirements and the elements of the model.

The proposed information model allows the expression of the handled concepts (requirements, design solution and V&V), and the creation of traceability links between these concepts in order to facilitate the comprehension and/or the impacts analysis. The proposed approach formalizes the practice of systems engineering through the use of models. The objective is to improve quality/productivity and to reduce risk by introducing rigor, precision, and communications among system/project stakeholders and managing complexity.

The approach is demonstrated through an example from aeronautic domain and presented in (Guillerm, 2011). However, the approach needs an additional work for its validation and application.

Future work will consist on considering other EIA-632 processes. Indeed in the present paper we considered only the processes of system design and evaluation. The consideration of processes like technical management or acquisition and supply will complete the integration approach.

With regard to information model, we are studying the interest of using OCL (object constraint language) in order to formalize the requirements. The objective is to integrate this possibility in the information model. Another point concerns the definition of some analysis from the information model, such as the generation of traceability matrix.

References

(Aer, 2004) SAE Aerospace. SAE *AS5506 : Architecture Analysis and Design Language (AADL)*. SAE International, 2004.

(AFIS, 2005) AFIS, *Modèle de données AFIS, version 2.0*, groupe de travail Méthodes et Outils, 2005.

(Akerlund , 2006) Akerlund. O. *ISAAC, a framework for integrated safety analysis of functional, geometrical and human aspects*. European Congress on Embedded Real-Time Software, Toulouse, 25, 26, 27/01/06

(Albinet and al, 2008) Albinet. A, Begoc. S, Boulanger. J-L, Casse. O, Dal. I, Dubois. H, Lakhal. F, Louar. D, Peraldi-Frati. M-A, Sorel. Y, Van. Q-D. *The MeMVaTEx methodology: from requirements to models in automotive application design*. International Conference : ERTS EMBEDDED REAL TIME SOFTWARE 2008 TOULOUSE, FRANCE, January 29, 30, 31, February 1, 2008

(ARP-4754, 1996) ARP 4754, *Certification considerations for Highly-Integrated or Complex Aircrafts Systems*, SAE, 1996.

(ARP-4761, 1996) ARP-4761, *Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment*, SAE, 1996.

(Auyang, 2004) Auyang, S. Y., *Engineering – An Endless Frontier*, Cambridge, MA, Harvard University Press, 2004.

(Avizienis and al, 2004) Avizienis. A, J.-C. Laprie, B. Randell, and C. Landwehr. *Basic Concepts and Taxonomy of Dependable and Secure Computing*. IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004.

(Bar-yam, 2005) BAR-YAM Yaneer. *About engineering complex systems: Multiscale analysis and evolutionary engineering*. Engineering self-organizing systems: methodologies and applications 2005, vol. 3464, pp. 16-3. ISBN 3-540-26180-X

(Black and Koopman, 2009) Black. J, Koopman. P. *System Safety as an Emergent Property in Composite Systems*. IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 369 - 378. Estoril, Lisbon, June 29- July 2 2009.

(Bock, 2006) Bock, C, *SysML and UML 2 Support for Activity Modeling*, Systems Engineering, 9, No. 2, pp. 160-186, 2006.

(Booch and al, 1998) Booch. G, Rumbaugh. J and Jacobson. I, *The Unified Modeling Language User Guide*, Addison- Wesley, 1998.

(Bozzano and al, 2003) Bozzano M, Cavallo. A, Cifaldi. M, Valacca. L, Villafiorita. A. *Improving Safety Assessment of Complex Systems: An Industrial case study*. FM 2003. Pisa, 8-14 September 2003

(Brooks, 1987) Brooks, F.P. Jr. *No Silver Bullet: Essence and Accidents of Software Engineering*. IEEE Computer, 10-19, April 1987.

(Conquet, 2008) Conquet E., The ASSERT project: a step towards reliable and scientific system and software engineering, ERTS2008, Toulouse, France, January 2008.

(Coulin and al, 2005) C.COULIN, A.E.K.SAHRAOUI, D.ZOWGHI, *Towards a collaborative and combinational approach to requirements elicitation within a systems engineering framework.* 18th International Conference on Systems Engineering (ICEng'05), Las Vegas (USA), 16-18 Août 2005, pp.456-4.

(EIA-632, 1999) *EIA-632: Processes for engineering systems*, Electronic Industries Alliance standard, 7 janvier 1999.

(Friedenthal and Kobryn, 2004) Friedenthal. S and Kobryn. C, *Extending UML to support a systems modeling language*, INCOSE Symp Proc, Toulouse, France, June 2004.

(Goguen and Linde, 1993) Goguen. J and C. Linde. *Techniques for requirements elicitation. In 1st IEEE International Symposium on Requirements Engineering*, pages 152-164, San Diego, 4-6th January 1993.

(Gotel and Finkelstein, 1994) Gotel. O. C. Z. and C. W. Finkelstein, *An analysis of the requirements traceability problem*, in International Conference on Requirements Engineering, 1994, pp. 94– 101.

(Guillerm and al, 2010) Guillerm. R, Sadou. N , Demmou. H., *Information model for model driven design of complex system based on system engineering approach*, International Conference onComplex Systems Design and Management (CSDM 2010), Paris (France), 27-29 Octobre 2010, pp.99-111.

(Guillerm, 2011) Guillerm. R *Intégration de la Sûreté de Fonctionnement dans les Processus d'Ingénierie Système*, PhD Thesis, university of Toulouse, 2011.

(Juristo and al, 2002) Juristo. N, Moreno. A. M, and Silva. A, *Is the European Industry Moving Toward Solving Requirements Engineering Problems?* IEEE Software, vol. 19, no. 6, pp. 70–77, 2002.

(Kotovsky and al 1985) Kotovsky. K, Hayes. J.R, and Simon. H.A. *Why are some problems hard? Evidence from Tower of Hanoi.* Cognitive Psychology, vol. 17, 1985.

(Levenson, 2004) Leveson. N.G. *Model-based analysis of socio-technical risk.* Technical Report ESD-WP-2004-08, MIT, Cambridge, MA, Dec. 2004.

(Leveson & al., 2007) Leveson N., M. S. Herring, B. D. Owens, M. Ingham et K. A. Weiss, Safety-Driven Model-Based System Engineering Methodology Part I: Methodology Description, 16 decembre 2007.

(Magee and de Weck, 2004) Magee C. and de Weck O. L., *Complex System Classification*, Fourteenth Annual International Symposium of the International Council on Systems Engineering (INCOSE), Toulouse, France, June 20-24, 2004.

(MIL-STD-499A, 1969) MIL-STD-499A, *Engineering Management*, 1969.

(Parviainen and al, 2004) Parviainen. P, M. Tihinen, Lormans. M, and Van Solingen. R, *Requirements Engineering: Dealing with the Complexity of Sociotechnical Systems Development*, in Requirements Engineering for Sociotechnical Systems, J. L. Mat´e and A. Silva, Eds. IdeaGroup Inc, 2004, ch. 1, pp. 1–20.

(Rasmussen, 1997) Rasmussen J., *Risk Management in a Dynamic Society: A Modelling Problem*, Safety Science, vol. 27, No. 2/3, Elsevier Science Ltd., 1997, pp. 183213.

(Roberston, 2010) Robertson J. and S., *Volere Requirements Specification Template*, edition 15, Atlantic Systems Guild, 2010.

(Robertson and Robertson, 2006) Robertson. S and J. Robertson, *Mastering the Requirements Process (2nd Edition)*. Addison-Wesley Professional, 2006.

(Sahraoui and al 2004) Sahraoui. A.E.K, Buede. D, Sage. A, *Issues in systems engineering research*, INCOSE congress, Toulouse, 2004.

(Sahraoui, 2005) Sahraoui. A.-E.-K., *Requirements Traceability Issues: Generic Model, Methodology And Formal Basis*. International Journal of Information Technology and Decision Making, vol. 4, no. 1, pp. 59–80, 2005.

(Sanford and al, 2009) Sanford Friedenthal, Alan Moore and Rick Steiner. *A Practical Guide to SysML The Systems Modeling Language*. 576 The MK/OMG Press Series 2009)

(Sommerville, 2006) Sommerville. I, *Software Engineering: (Update) (8th Edition)* (International Computer Science). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006.

(SysML, 2006) SysML Merge Team, *Systems Modeling Language (SysML) Specification*, http://www.omg.org/cgi-bin/doc?ad/06- 02-01, February 2006.

(Valerdi and Wheaton, 2005) Ricardo Valerdi , Marilee Wheaton, *ANSI/EIA 632 As a Standardized WBS for COSYSMO*. AIAA 5th Aviation, Technology, Integration, and Operations Conference (ATIO) 26 - 28 September 2005, Arlington, Virginia.

(Verries and al, 2008) J.Verries, M. Paludetto, A-E-K. Saharaoui. *From design with SysML to VHDL-AMS simulation*, Proceeding of ESM'08, Le Havre, Oct. 2008, pp.115-120.),

(Vhd, 1999) 1076.1-1999 *IEEE Standard VHDL Analog and Mixed-Signal Extensions Language Reference Manual*, IEEE Press, ISBN 0-7381-1640-8.

(Wu and Kelly, 2006) Wu. W, Kelly. T.P., *Deriving Safety Requirements as Part of System Architecture Definition*. In Proceedings of 24th International System Safety Conference, published by the System Safety Society, August 2006, Albuquerque, USA.

(Zeigler and al, 2000) Zeigler. B.P, Praehofer. H and Kim. T.G., *Theory of modelling and simulation*. Academic Press, San Diego, California, USA, 2000.